CIS 520 Machine Learning Summary

Lyle Ungar

What we covered
What’s hot in ML

Final: Tuesday 12/20 6:00 pm
Watch ed for location
Towne 100 / Wu & Chen
Two 2-sided pages cheatsheet
Course goals

◆ Be familiar with all major ML methods
  ● Regression (linear, logistic), regularization, feature selection
  ● K-NN, Decision trees, Random Forests, SVMs
  ● PCA, K-means, GMM, Autoencoders
  ● Naive Bayes, Bayes Nets, LDA, HMMs
  ● Boosting, perceptrons, LMS
  ● Deep learning (CNNs)
  ● Reinforcement Learning (MDP, Q-learning)

◆ Know their strengths and weaknesses
  ● know jargon, concepts, theory
  ● be able to modify and code algorithms
  ● be able to read current literature

We did all of these!
Components of ML

◆ Representation
  ● Feature set
  ● Model form

◆ Loss function
  ● And regularization penalty

◆ Optimization method
  ● For parameter estimation
  ● For model selection and hyperparameter tuning
Representations

- **Non-parametric**
  - Nearest-neighbor
  - Decision Trees, Random forests, gradient tree boosting

- **Linear models**
  - Hyperplane as a separator
  - Kernel methods

- **Neural nets**
  - CNN’s, Recurrent Nets/LSTMs

- **Belief nets**
  - HMM, LDA
## Representations

<table>
<thead>
<tr>
<th>Linear (parametric)</th>
<th>Nonlinear (semi-parametric)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OLS</td>
<td>Neural Nets</td>
</tr>
<tr>
<td>Logistic regression</td>
<td>Nonlinear (nonparametric)</td>
</tr>
<tr>
<td>HMM</td>
<td>K-NN</td>
</tr>
<tr>
<td>MDP</td>
<td>Trees, Forests</td>
</tr>
</tbody>
</table>
## MLE gives loss functions

<table>
<thead>
<tr>
<th>Loss function</th>
<th>Bayesian (MLE/MAP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OLS</td>
<td>OLS</td>
</tr>
<tr>
<td>K-means</td>
<td>GMM</td>
</tr>
<tr>
<td>PCA</td>
<td></td>
</tr>
</tbody>
</table>

Gaussian noise gives L2 loss
Representations: Primal/Dual

**Primal**: feature space
- $X^TX$
- Covariance
- OLS

**Dual**: observation space
- $XX^T$
- Kernel matrix
- SVM
Invariances

Translational invariance

**In space:** CNN, data augmentation

**In time:** CNN, HMM, MDP, RNN
What loss functions have we used?

- L0, L1, L2
- Log-likelihood (MLE, MAP)
- Hinge
- Logistic
- Exponential
- Cross-Entropy; KL-divergence

Boosting:  \( \exp(-y_i f_\alpha(x_i)) \)  Logistic:  \( \log(1 + \exp(-y_i f_w(x_i))) \)
Loss Functions

- $L_0$
- Hinge
- Logistic
- Exponential (adaboost)
Regularization priors

\[
\text{Argmin}_w \|y - w \cdot x\|_2^2 + \lambda \|w\|_p^p
\]

- **L_2** \( \|w\|_2^2 \)
  - Gaussian prior: \( p(w) \sim \exp(-|w|_2^2/\sigma^2) \)

- **L_1** \( \|w\|_1 \)
  - Laplace prior: roughly \( p(w) \sim \exp(-|w|_1/\sigma^2) \)

- **L_0** \( \|w\|_0 \)
  - Spike and slab prior

\[
\log P(D_X, D_Y, \theta) = \log P(D_X, D_Y \mid \theta) + \log P(\theta) = -\text{loss}(\theta) + \text{regularizer}(\theta)
\]
Bias-Variance Trade-off

\[ E_{x,y,D}[(h(x; D) - y)^2] = \]

\[ \underbrace{E_{x,D}[(h(x; D) - \bar{h}(x))^2]}_{\text{Variance}} + \underbrace{E_x[(\bar{h}(x) - \bar{y}(x))^2]}_{\text{Bias}^2} + \underbrace{E_{x,y}[(\bar{y}(x) - y)^2]}_{\text{Noise}} \]
Optimization methods

- **Closed form** (e.g. \( w = (X^TX)^{-1}X^T y \))
- **Gradient descent**: Stochastic, minibatch
  - Streaming/Online: LMS, Perceptron
- **Search**: streamwise, stepwise, stagewise
- **Power method** (for eigenvectors, SVD)
- **Lagrange Multipliers** (constrained optimization)
  - not covered
Alternating optimization methods

◆ **EM** (alternating gradient descent in likelihood)
  - E: expected value of hidden values
  - M: MLE or MAP estimate of parameters

◆ **Other alternating methods**
  - X \sim SW^T  for ICA, NNMF (non-negative matrix factorization)
  - RL: V or Q and policy
    - **Response surface**: Model and optimal action
Hyperparameter Optimization

◆ **Search**
  - e.g., $L_1$, $L_2$, penalties
  - Neural network structure, regularization

◆ **Auto-SKlearn**
  - Initialize hyperparameters from model predicting accuracy as a function of problem description and hyperparameter values

◆ **Auto-ML**
  - Use reinforcement learning to learn a ‘design policy’
Distance and similarity

◆ Distances from norms
  • $||x_1-x_2||_0$  $||x_1-x_2||_1$  $||x_1-x_2||_2$  ...

◆ Similarities from kernels
  • $k(x_1,x_2)$

◆ Probability-based divergence
  • $D_{KL}(p||q) = \sum_k p_k \log(p_k/q_k)$  - KL-divergence
  • $H(p,q) = H(p) + D_{KL}(p||q)$  - cross-entropy

  \[= - \sum_k p_k \log(q_k)\]
  - $p$ is the true distribution, $q$ is the approximation
Cross entropy and log-likelihood

- **Cross-entropy**
  
  - $H(p,q) = - \sum_k p_k \log(q_k)$ summed over labels $k$
  
  - $- \sum_i \sum_k \delta_{ik} \log(p(y_i=k|x=x_i))$  \hspace{1em} $\delta_{ik} = 1 \text{ iff } y_i = k$
    
    - Sum of the estimated log probabilities of the true answers
  
- $\log \prod_i p(y_i|x_i) = \sum_i \log p(y_i|x_i)$ log-likelihood
KL-Divergence

- $D_{KL}(p\|q) = \sum_k p_k \log(p_k/q_k)$

- **Mutual information** – not really covered
  - $MI(X,Y) = D_{KL}(P(X,Y) \| P(X)P(Y))$

- **Information gain**
  - $IG(Y|X_j) = D_{KL}(P(Y|X_j) \| P(Y)) = H(Y) - H(Y|X_j)$
  - Which feature $X_j$ will maximize the information gain?

- **Bayesian Experimental Design**
  - For which $x$ will the label $y$ (in expectation) most change $p(w)$

https://en.wikipedia.org/wiki/Kullback%E2%80%93Leibler_divergence
Types of Learning

- **Supervised** $X, y$
  - Given an observation $x$, what is the best label $y$?

- **Unsupervised** $X$
  - Given a set of $x$’s, cluster or summarize them

- **Reinforcement**
  - Given a sequence of states $x$ and possible actions $a$, learn which actions maximize reward.

What kind of learning is missing here?
Unsupervised methods

- PCA, ICA, NNMF
  - $X \sim S V^T$
- K-means, GMM, LDA
- Auto-encoders
  - Information bottleneck
  - Denoising
  - Variational

Many of these minimize reconstruction error subject to some constraints
Bayesian Belief Nets

- **Naïve Bayes**
  - Binary or real-valued X’s;
- **Belief Net**
- **GMM**
  - Different model forms
- **LDA**
- **HMM/MDP**
Reinforcement learning

◆ Model-based
  ● MDP

◆ Model-free
  ● Shallow: TD(0) vs. Deep: Monte-Carlo Tree Search
  ● Value: \( V(s) \) vs. Q-learning \( Q(s,a) \)

◆ On-policy (\( \epsilon \)-greedy) vs. off-policy
  ● Trade-off exploration and exploitation
Summary

Response to all possible actions

Response to one possible actions

One-step ahead

Search to end

Model-based

Model-free

From David Silver UCL Course on RL: http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html
For any MDP, given infinite exploration time and a partly-random policy, Q-learning will find an optimal policy: one that maximizes the expected value of the total reward over all successive steps.

\[
Q^{\text{new}}(s_t, a_t) \leftarrow (1 - \alpha) \cdot Q(s_t, a_t) + \alpha \cdot \left( r_t + \gamma \cdot \max_a Q(s_{t+1}, a) \right)
\]
Deep Q-Learning (DQL)

$$\text{Argmin}_\theta \left[ Q(s, a; \theta) - \left( r(s, a) + \gamma \max_a Q(s', a; \theta) \right) \right]^2$$

Represent $Q$ with a neural net

$s, a$ can be one-hot or real valued

Update this

To be closer to new value estimate
What to use when?
SKLearn vs. NNets

◆ Deep learning is almost always better than classic ML on large data sets
  - Text, images, sound, videos

◆ Classic ML is often better than deep learning on tabular data
Feature selection

- **Regression** (L0, L1, L2 penalties)
  - Do you expect very few, a moderate number of, or most features?

- **Random forests, gradient tree boosting**
  - Feature selection is ‘built in’

- **Neural nets**
  - Generally, no built-in feature selection
  - Screen features before you build the net
Note:

- The new material after this slide will not be on the final; it is just for fun!
What’s hot

◆ Applied ML
  ● datascience
◆ Multimodal
◆ Human in the loop
◆ Generative models
  ● Stable diffusion
  ● ChatGPT

This is a photograph of ancient Greek philosopher Heraclitus in 500 BC.
What’s hot: generative models

- Given a set of observations, $x$, generate new $x$’s from the same distribution

- Diffusion Models
  - $p(\text{image}' \mid \text{words}, \text{image})$

- Large Language Models
  - $p(\text{word}_{t+1} \mid \text{word}_t, \text{word}_{t-1}, \text{word}_{t-2}, ...)$
Diffusion Models

Diffusion Models

◆ Dall-E 2, Stable Diffusion, Midjourny

Source: https://www.youtube.com/watch?v=F1X4fHzF4mQ
Diffusion Models

Markov Chain!

https://medium.com/@monadsblog/diffusion-models-4dbe58489a2f
Large Language Models

- **GPT-3, ChatGPT** - OpenAI
- **Blenderbot** - Facebook
- **PaLM, Lambda** - Google
GPT-3 Generative Pretrained Transformer

► Trained to predict next word
  ● on ~ 45TB of text

► ~ 175B parameters.

► 2048 token context
  ● About 1,500 words

► 96 transformer layers

► GPT-4 will have 100 Trillion parameters
Transformers

- **Encoder-decoder architecture**
  - With self-attention: learns how much weight to put on each token

- **Byte Pair Encoding (BPE) tokenization**
Self-attention

- Embed every token in the sentence
- Project them down to \( Q, K, V \)
- Reweight them with \( \text{softmax}(Q K^T) \)
- Do this many times (different "heads")

https://jalammar.github.io/illustrated-transformer/
Self-attention

Embed every token in the sentence
Project them down to $Q$, $K$, $V$
Reweight them with $\text{softmax}(QK^T)$
Do this many times (different “heads’)

https://jalammar.github.io/illustrated-transformer/
# ChatGPT

<table>
<thead>
<tr>
<th>Examples</th>
<th>Capabilities</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Explain quantum computing in simple terms&quot; →</td>
<td>Remembers what user said earlier in the conversation</td>
<td>May occasionally generate incorrect information</td>
</tr>
<tr>
<td>&quot;Got any creative ideas for a 10 year old’s birthday?&quot; →</td>
<td>Allows user to provide follow-up corrections</td>
<td>May occasionally produce harmful instructions or biased content</td>
</tr>
<tr>
<td>&quot;How do I make an HTTP request in Javascript?&quot; →</td>
<td>Trained to decline inappropriate requests</td>
<td>Limited knowledge of world and events after 2021</td>
</tr>
</tbody>
</table>
See all of you for the final, **Tuesday 6:00**

Stay in touch & let me know how you use ML …

Thank you!!!